Self-consistency and MDL: a Paradigm for Evaluating Point Correspondence Algorithms and Detecting Change

Yvan G. Leclerc, Q.-Tuan Luong and P. Fua *
Artificial Intelligence Center, SRI International, Menlo Park, CA
leclerc,luong@ai.sri.com
LIG, EPFL, Lausanne, Switzerland
fua@lig.di.epfl.ch

October 16, 2000

Abstract

Self-consistency is a new paradigm for evaluating vision algorithms without relying extensively on ground truth. We demonstrate its effectiveness in the case of point-correspondence algorithms and use our approach to predict their accuracy.

Our methodology consists of applying independently the point correspondence algorithm to subsets of images obtained by varying the camera geometry while keeping 3-D object geometry constant. Matches which should correspond to the same surface element in 3-D are collected to create statistics, that are then used to infer accuracy and reliability.

An effective representation for this is a scatter diagram along two dimensions: a matching score and a normalized distance. We introduce a new matching score based on MDL theory, which is shown to be a better predictor of the quality of a match than the traditional SSD score. The normalization of the distance is shown to make the statistics invariant to camera geometry.

We demonstrate the potential of the methodology in two different application areas. First, we compare different point correspondence algorithms, matching scores, and window sizes. Second, we detect change in shape between 3-D models reconstructed from two sets of images taken at a different time.

We finish by discussing the application of self-consistency to more general vision problems.

1 Introduction

Our visual system has a truly remarkable property: given a static natural scene, the perceptual inferences it makes from one viewpoint are almost always *consistent* with the inferences it makes from a different viewpoint. We call this property *self-consistency*. The ultimate goal of our research is be able to design computer vision algorithms that are also self-consistent. The first step towards achieving this goal is to measure the self-consistency of the inferences of current computer vision algorithm over many scenes. In this paper, we focus on point correspondence (or "stereo") algorithms, which constitutes one of the most important classes of computer vision algorithms.

*This work was sponsored in part by the Defense Advanced Research Projects Agency under contract F33615-97-C-1023 monitored by Wright Laboratory. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency, the United States Government, or SRI International.

One possible statistical characterization of the performance of a point-correspondence algorithm is in terms of the error of matches compared to "ground truth." If sufficient quantities of accurate ground truth were available, estimating the distribution of errors over many image pairs of many scenes would be relatively straightforward. This distribution could then be used as a prediction of the accuracy of matches in new images. Unfortunately, acquiring ground truth for many scenes is an expensive and problematic proposition at best. Instead, we propose to estimate a related distribution, the *self-consistency distribution* which can be derived automatically from the matches of many image pairs of many scenes. All the additional information we need is the precise estimation of the camera parameters (projection matrices), and ideally of their covariances.

Assuming that bias have been removed, a task which can be done with a minimal amount of ground truth data, the self-consistency distribution gives a global characterization of the accuracy of a stereo algorithm.

To go further, and be able to predict the accuracy of individual matches, we would like to attach a number to each match, a "score" that would be a reliable predictor of the accuracy of that match. Although there has been much work in predicting the accuracy of a match based on scores such as correlation, these theoretical predictions can only be derived for very simple scene geometries and imaging conditions. One of the difficulties is that the scores which are traditionally used, such as the sum of squared differences (SSD) are ambiguous. To solve this problem, we develop a new scoring mechanism based on Minimum Description Length (MDL) theory. This new measure is designed to be a better indicator of the quality of a match than the SSD score. Using extensive experiments, we show that, indeed, it correlates with the self-consistency of matches better than any other score we are aware of.

The self-consistency distribution is a very simple idea that has powerful consequences beyond the applications described in this paper. It can be used to compare algorithms, compare scoring functions, evaluate the performance of an algorithm across different classes of scenes, tune algorithm parameters, and so forth. Detecting where an object's 3-D shape has changed requires not only the ability to model shape from images taken at different times, but also the ability to distinguish significant from insignificant differences due variability in the models derived from two image sets. Since self-consistency excels at predicting expected variability, a natural application of the self-consistency methodology is change detection.

The three next sections introduce the components of the self-consistency methodology applied to stereo or multiple image correspondence algorithms. In Sec. 2, we introduce a new kind of distribution, meant to characterize self-consistency. Two important aspects are the conditionalization with respect to an appropriate score, and a proper normalization. In Sec. 3, such a score is developed, as a better alternative to the traditional SSD measure, and in Sec. 4, it is explained how the normalization makes the self-consistency measures invariant with respect to the geometry of the projection. The usefulness of self-consistency is illustrated in the two following sections. In Sec. 5, we apply measurements of the self-consistency of some stereo algorithms to a variety of real images to demonstrate the utility of these measurements for comparing algorithms. In Sec. 6 we apply self-consistency measurements to the problem of detecting change in shape between two sets of images taken at a different time. The three last sections place self-consistency in a broader perspective. In Sec. 7, we compare our approach to previous work on measuring uncertainty and detecting change. In Sec. 8, we show how the self-consistency paradigm can be used for several vision algorithms, beyond stereo. We conclude in Sec. 9 and point to interesting possibilities opened by the new formalism.

2 Self-Consistency of Stereo algorithms

2.1 The basic idea

We start with a fixed collection of images taken at exactly the same time (or, equivalently, a collection of images of a static scene taken over time). Each image has a unique index and

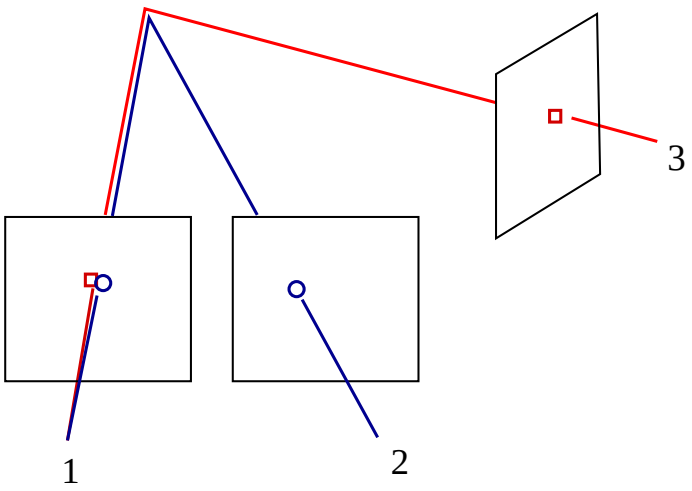


Figure 1: Two matches which have a common point in one image should correspond to the same point in the world

associated projection matrix and (optionally) projection covariances, which are supposed to be known. This projection matrix describes the projective linear relationship between the 3-D coordinates of a point in a common coordinate system, and its projection in the image. Although the methodology is easier to apply when the common coordinate system is Euclidean, the minimal requirement is that the set of projection matrices be a common projective coordinate system. This could be obtained from point correspondences using projective bundle adjustment [21, 29] and does not require camera calibration.

We then apply a point correspondence algorithm, (which we will designate in the remainder of the paper by "stereo" algorithm for brevity) independently to all pairs (or n-tuples) of images in this collection. For this paper, a stereo algorithm takes as input a set of N images of a stationary scene and associated projection matrices $\mathbf{P}_i$, $i = 1 \ldots N, N > 2$. The output of a stereo algorithm is a set of matches of 2-D points and, optionally, a score s_j, which represents a measure of the algorithm's confidence in the corresponding match. The score would have a low value when the match is certain, and a high value when the match is uncertain. Each match has between 2 and N 2-D points which are supposed to represent the 2-D projections of a single point in the world.

The image indices, match coordinates, and score, are reported in *match* files for each image pair. We now search the match files for pairs of matches that have the same coordinate in one image. For example, if a match is derived from images 1 and 2, another match is derived from images 1 and 3, and these two matches have the same coordinate in image 1, then these two matches, which we call a *common-point match set*, should be self-consistent because they should correspond to the same point in the world, as illustrated in Fig. 1. This extends the principle of the trinocular stereo constraint [32, 3] to arbitrary camera configurations and multiple images.

If the stereo algorithm provides the covariances[1] of the matches, we can derive the probability that a pair of matches has a common point, and take that into account in the self-consistency statistics below by thresholding or weighting. Otherwise, we could assume a nominal accuracy (typically 1 pixel) for the precision of localization.

Given two matches in a common-point match set, we can now compute the distance between their triangulations, after normalizing for the camera configurations (see Sec. 4). The histogram of these normalized differences, computed over all common-point matches, is our estimate of the self-consistency distribution. This distribution is meant to characterize the accuracy of the

[1] A measure of the localization of the match, as opposed to the score defined above, a measure of the confidence of the match.

particular stereo algorithm applied to the particular set of images.

Self-consistency is a necessary, but not sufficient, condition for a computer vision algorithm to be correct. That is, it is possible (in principle) for a computer vision algorithm to be self-consistent over many scenes but be severely biased or entirely wrong. We conjecture that this cannot be the case for non-trivial algorithms. If bias can be ruled out, then the self-consistency distribution becomes a measure of the accuracy of an algorithm—one which requires no "ground truth." In practice, to rule out bias, it is sufficient to have ground truth only for a few scenes.

2.2 An Example of the Self-Consistency Distribution

To illustrate the self-consistency distribution, we first apply the above methodology to the output of a simple stereo algorithm [10]. The algorithm first rectifies the input pair of images and then searches for 7×7 windows along scan lines that maximize a normalized cross-correlation metric. Sub-pixel accuracy is achieved by fitting a quadratic to the metric evaluated at the pixel and its two adjacent neighbors. The algorithm first computes the match by comparing the left image against the right and then comparing the right image against the left. Matches that are not consistent between the two searches are eliminated. Note that this is a way of using self-consistency as a filter.

The stereo algorithm was applied to all pairs of five aerial images of bare terrain, one of which is illustrated in the top row of Figure 2(a). These images are actually small windows from much larger images (about 9000 pixels on a side) for which precise ground control and bundle adjustment were applied to get accurate camera parameters.

Because the scene consists of bare, relatively smooth, terrain with little vegetation, we would expect the stereo algorithm described above to perform well. This expectation is confirmed anecdotally by visually inspecting the matches.

However, we can get a quantitative estimate for the accuracy of the algorithm for this scene by computing the self-consistency distribution of the output of the algorithm applied to the ten images pairs in this collection. Figure 2(b) shows two versions of the distribution. The solid curve is the probability density (the probability that the normalized distance equals x). It is useful for seeing the mode and the general shape of the distribution. The dashed curve is the cumulative probability distribution (the probability that the normalized distance is less than x). It is useful for seeing the median of the distribution (the point where the curve reaches 0.5) or the fraction of match pairs with normalized distances exceeding some value.

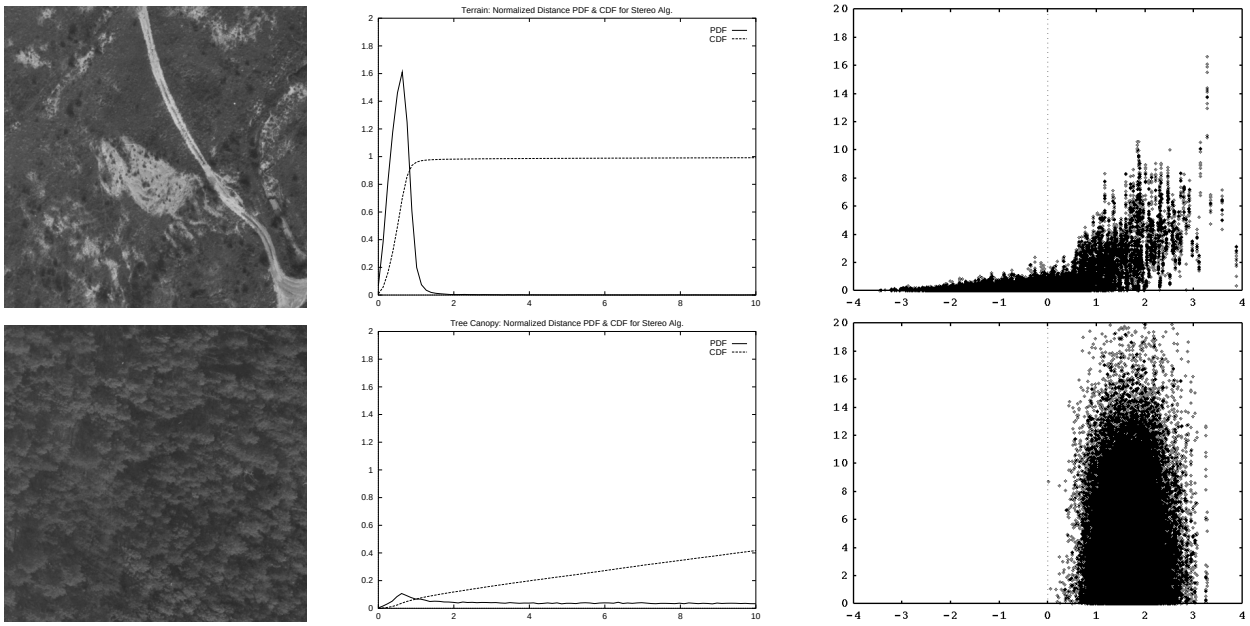
In this example, the self-consistency distribution shows that the mode is about 0.5, about 95% of the normalized distances are below 1, and that about 2% of the match pairs have normalized distances above 10.

In the bottom row of Figure 2 we see the self-consistency distribution for the same algorithm applied to all pairs of five aerial images of a tree canopy. Such scenes are notoriously difficult for stereo algorithms. Visual inspection of the output of the stereo algorithm confirms that most matches are quite wrong. This can be quantified using the self-consistency distribution in Figure 2(b). Here we see that, although the mode of the distribution is still about 0.5, only 10% of the matches have a normalized distance less than 1, and only 42% of the matches have a normalized distance less than 10.

Note that the distributions illustrated above are not well modelled using Gaussian distributions because of the predominance of outliers (especially in the tree canopy example). This is why we have chosen to compute the full distribution rather than use its variance as a summary.

2.3 Conditionalization

The self-consistency distribution presented so far is an estimate of the overall precision of the algorithm applied to a specific set of images. However, often it is desirable to have a finer and also more general estimation of the precision. First, individual output surface elements do not have the same precision. We would like to be able to characterize that. Second, from the previous examples, the self-consistency distribution varied considerably from one scene to



(a) sample image (b) self-consistency distributions (c) scatter diagrams with MDL score

Figure 2: Self-consistency distributions and score-dependent diagrams for two different types of images: terrain (top) vs. tree canopy (bottom).

the next. We would like to be able to characterize the performance of the algorithm applied to a wide variety of scenes, in a way which would be more independent from the particular scenes chosen. These goals are achieved by (a) computing self-consistency as a function of an appropriate score, (b) computing self-consistency distributions as a function of a class of scenes and a particular algorithm.

The self-consistency scatter diagram So far, all the matches were treated uniformly, since there was no particular information attached to them. To go further, the idea is to segregate the matches according to a particular number. This number is the "score" associated with each match. It is an indication of the confidence of the algorithm in the match. It has a low value when the match is certain, and a high value when the match is uncertain.

Using the score, we can create a new representation, called the *self-consistency scatter diagram*. An example is in Figure 2(c). This scatter diagram shows a point for every pair of common-point matches, the x coordinate of the point being the larger of the scores of the two matches, and the y coordinate being the normalized distance between the matches.

For practical purposes such as algorithm comparison and change detection, the information in the scatter diagram can be summarized by a series of curves, called *s% significance level curves* $f(s, x)$. For a given value of the score x , the s% significance level is the largest normalized difference for which s% of the common-point matches lie below: $P(x < f(s, x)) = s$. Examples of significance level curves can be seen in Fig. 7.

There are several points to note about the previous example. First, the terrain example (top row) shows that most points with scores below 0 have normalized distances less than about 1. Second, most of the points in the tree canopy example (bottom row) are not self-consistent. Third, none of the points in the tree canopy example have scores below 0. Thus, it would seem that this score is able to segregate self-consistent matches from non-self-consistent matches, even when the scenes are radically different. An ideal score would have the two following properties:

- The relationship between the score and the degree of self-consistency is linear, therefore

the significance level curves are linear. In practice it would be enough if the relationship was monotonic over the "good" (ie low) scores. For the "bad" (high) scores, in most applications it is sufficient to know that the algorithm failed to obtain a satisfactory match. Looking at the top diagram of Fig. 2(c), it seems that for the negative values of the score, the relationship is approximatively linear.

- Matches with the same score have the same degree of self-consistency. The significance level curves would be constant across any sets of images, except for the scores for which there is not enough data to support a meaningful computation of the significance level, like for the negative values of the score in the bottom diagram of Fig. 2(c). In practice, it would be enough if the property only hold statistically, and the curves were constant only across a representative class of images, and this will be discussed in the next paragraph.

Having a score which satisfies these two requirements makes it possible to predict the self-consistency, and therefore accuracy, of the reconstruction of a 3D surface element just by looking up the score(s) associated with the matches used to reconstruct it. In Sec. 5.2, we will compare several different scores to determine experimentally which ones satisfy these two requirements best.

Aggregating the scatter diagrams From the diagrams in the previous example, it is not clear at first that the second requirement is satisfied, since the two diagrams appear to be very different. This is because the two types of scenes are so different that there is little overlap in terms of score. However, look at the left diagram of Fig. 8. This diagram was obtained by merging results from seventeen scenes. The scatter diagram is a representation of self-consistency which makes it particularly convenient to merge several distributions: all is needed to do is to take the union of the points which compose the diagrams.

One can notice that for negative values of the score, the diagram looks quite similar to the top diagram in Fig. 2(c), while for positive values of the score, it looks quite similar to the bottom diagram in Fig. 2(c). In fact, two collections of scenes with a different mixture of the two types would yield an identical aggregate scatter diagram. If, instead of using the scatter diagram, we aggregated the self-consistency distributions (ie did not segragate by score), the result would depend on the mixture used.

Ideally, the self-consistency diagrams should be computed using all possible variations of viewpoint and camera parameters (within some class of variations) over all possible scenes (within some class of scenes). However, we can compute an estimate of the diagram using some small number of images of a scene, and aggregate the diagrams over many scenes. As the number of scenes increases, the diagram hopefully converges towards a certain value. It is entirely possible that if we tried to aggregate all the possible types of scenes, there wouldn't be any convergence. This would indicate a limitation of the scoring mechanism. To cope with this problem, we need to restrict sufficiently the class of scenes considered. The closer to ideal a score is, the less restrained would be the class of scenes for which convergence occurs. For example, we have verified experimentally that for the MDL score described in Sec.3, the convergence occurs for classes of scenes such as the set of images of the rural scenes or the set of images of the urban area shown in Sec. 6.

We can then use the aggregate of the diagrams to represent the self-consistency diagram of new images of a new scene within the same class. Having a score which is a good predictor of self-consistency makes it possible to use an aggregated self-consistency diagram of a class to predict the expected variation in reconstruction when we have only a single image pair of a new scene within the same class. Since self-consistency is correlated with the quality of reconstruction, we would hope that with a suitable score, the reconstructions will be similar for places where the match score was good and dissimilar otherwise.

2.4 A few variations

Triangulations and reprojections As discussed previously, to apply the self-consistency method to a set of images, all we need is the set of projection matrices in a common projective coordinate system. The Euclidean distance, which is used to compare two triangulations, is not invariant to the choice of projective coordinates, but this dependence can often be reduced by using the normalization described in Sec. 4. If the cameras are calibrated, the projection matrices are obtained in a common Euclidean coordinate system, so this problem would not arise.

A method to cancel the dependence on the choice of projective coordinates, is to compute the difference between the reprojections instead of the triangulations. For two matches that have a common point in image 1, one match derived from images 1 and 2, and one derived from images 1 and 3, we can triangulate the first match, and reproject it into image 3. We can then compare the image coordinates of the reprojected point against the coordinate of the second match in image 3. Similarly for image 1. The dependence on the arbitrary projective basis used in 3D is eliminated because the projections are independent of this basis, unlike the triangulations. This, however, does not cancel the dependence on the relative geometry of the cameras, which is dealt with in Sec. 4.

More details and results using those distributions were reported in [16]. In the calibrated case, we favor the comparison of the triangulated points over the comparison of the reprojected points because it is more symmetric and computationally more simple, but in the case of uncalibrated cameras, the reprojection method would be preferable.

Image centered vs object-centered techniques So far the description was for stereo methods which produce point matches. The above procedure can also be used for object-centered surface reconstruction techniques such as [11]. In this case, one needs to compare 3–D surface reconstructions derived from different image sets of a static scene. Again, this can be done either in image space or in Euclidean space. In the first case, one can simply derive a dense set of matches by casting a ray through every pixel of, say, image 1 into the world. If the ray intersects the reconstructed surface, then reproject it into all of the images used to derived the surface. This will provide a dense set of matches that can be used as above. The second case requires identifying points on the surface that *should* be the same. One example of this will be used in Sec. 6: the surface is expressed as $z = f(x, y)$. The (x,y) coordinates are fixed, and only the z coordinate varies. This is useful for deriving terrain elevation models from aerial images, where the ground (x,y) coordinates are fixed, but the elevation is unknown. In this case, instead of finding matches which have a common 2-D point, we find matches which triangulate to 3-D points which have the same (x,y) coordinates. The histogram of the differences between the coordinates for such matches (appropriately normalized) is what we call the *common-xy-coordinate* self-consistency distribution.

Alternatives measures of deviation Another distribution that one could compute using the same data files would involve using all the matches in a common-point match set, rather than just pairs of matches. For example, one might use the deviation of the triangulations from the mean of all triangulations within a set. This is problematic for several reasons.

First, there are often outliers within a set, making the mean triangulation less than useful. One might mitigate this by using a robust estimation of the mean. But this depends on various (more or less) arbitrary parameters of the robust estimator that could change the overall distribution.

Second, and perhaps more importantly, we see no way to extend the normalization used to eliminate the dependence on camera configurations, described in Sec. 4, to the case of multiple matches.

Third, we see no way of using the above variants of the self-consistency distribution for change detection.

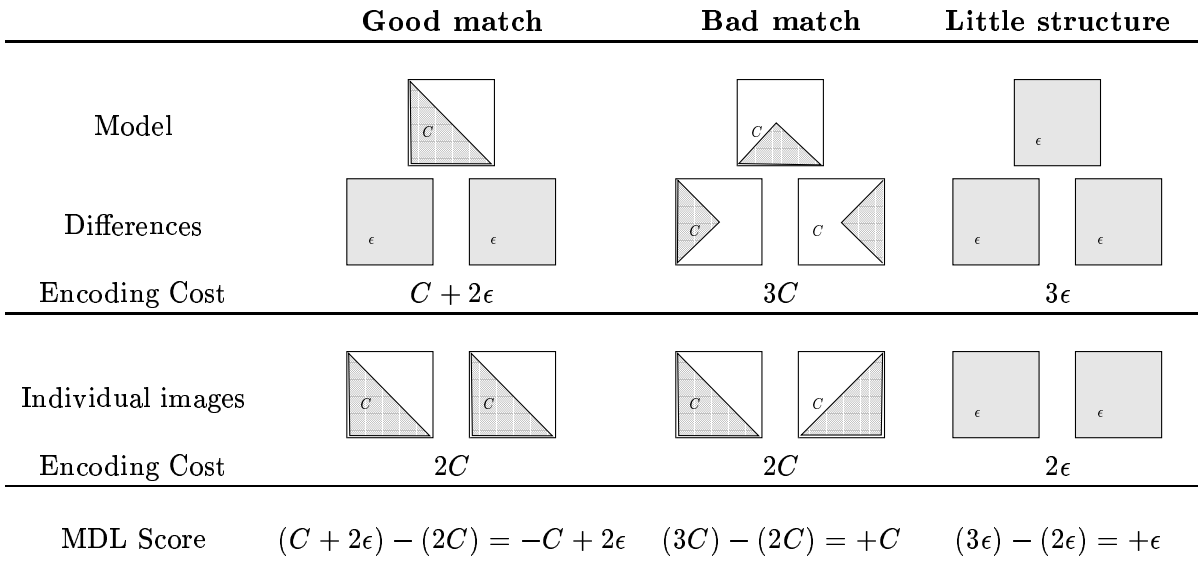


Figure 3: The MDL measure for a correct match with large pixel variation, incorrect match with large pixel variation, and small pixel value variation.

3 The MDL score

3.1 Principle of the coding loss

A classic score used in stereo is the sum of squared differences (SSD). The score is supposed to be small for "good" matches, and high for "bad" matches. The problem with the SSD measure is that it's ambiguous. That is, a low SSD measure can occur not only when the match is correct (as expected), but also when the match is incorrect and the terrain is spatially uniform (such as an unmarked parking lot, a flat sandy area, or a uniformly colored planar roof). Intuitively, then, we want an image-matching measure that is low only when the match between the predicted and observed pixel values is close and the pixel values form a sufficiently complex pattern that it is unlikely to be matched elsewhere.

We have developed a measure that satisfies this intuitive requirement reasonably well. We call this measure the coding loss. The coding loss is based on Minimum Description Length (MDL) theory [23]. In MDL theory, quantized observations of a random process are encoded using a model of that process. This model is typically divided into two components: a parameterized predictor function $M(\mathbf{z})$ and the residuals (differences) between the observations and the values predicted by that function. The residuals are typically encoded using an i.i.d. noise model [17]. MDL is a methodology for computing the parameters $\mathbf{z}$ that yield the optimal code length for this model and for a given encoding scheme. This optimal code length is the minimum number of bits required to encode the parameters $\mathbf{z}$ and the corresponding residuals, such that the resulting encoding is a loss-less encoding of the observations.

In our case, the process we are observing is the object surface. The observations are the pixel values in all of the images covered by a given surface element, which we call a facet. The parameters $\mathbf{z}$ are the coordinates and albedos of the facet. The parameterized predictor function $M(\mathbf{z})$ for the facet predicts the covered image pixel values given the facet parameters and the fixed camera parameters. The coding loss is the difference between the code length of the image pixels using the facet model and the code length using independent noise models. To see why this might be a good measure, consider the following cases, illustrated by Fig. 3.1:

1. *The facet is correct and the pixel value variation is large.* Here, we expect the residuals to be small (costing about ϵ bits to encode), and the material property samples to vary about

as much as the image pixels (costing, say, C bits to encode). Thus, the cost of encoding the image pixels using the facet model will be about $C+n\epsilon$ bits, while the cost of encoding the image pixels in the n images will be about nC bits. Thus, the coding loss is expected to be a large negative value: $C + n\epsilon - nC = (1-n)C + n\epsilon$ bits.

2. The facet is incorrect and the pixel value variation is large. Here, we expect the material property samples as well as the residuals to vary about as much as the image pixel values. Thus, the material property samples and the residuals for a given image will each cost about C bits to encode. Thus, the cost of encoding the image pixels using the facet model will be about $(n+1)C$ bits, while the cost of encoding the pixel values without the facet model will be about nC bits. Thus, the coding loss is expected to be a large positive value: $(n+1)C - nC = C$ bits.
3. The pixel value variation is small. Here, we expect the cost of encoding the property samples and the residuals to be about the same and small, say ϵ bits. Similarly, the cost of encoding the pixel values in a given image will be about ϵ bits. Thus, the coding loss is expected to be a small positive value: $(n+1)\epsilon - n\epsilon = \epsilon$ bits.

Note that cases 1 and 3 above are expected to have significantly different values for the coding loss, whereas they are expected to have about the same value for the SSD measure (namely, ϵ). Thus, we would expect (and have determined experimentally, see later) the coding loss to be more effective in distinguishing between correct and incorrect facet parameters. There are many ways of encoding both the facet parameters and the residuals. The choice of encoding schemes can have a significant effect on the overall code length and, more importantly, the values of the parameters that minimize the code length. To date we have used a very simple encoding scheme, which basically assumes that the parameters and residuals are well encoded using an i.i.d. white noise model. Even though this is clearly not an optimal encoding scheme, the results are very good. Better encoding schemes could significantly improve the results.

3.2 Analytical details

Given N images, let M be the number of pixels in the correlation window and let g_i^j be the image gray level of the i^{th} pixel observed in image j. For image j, the number of bits required to describe these gray levels as IID white noise can be approximated by:

$$C_j = M(\log \sigma_j + c) \tag{1}$$

where σ_j is the measured variance of the $g_{i\,1\le i\le N}^j$ and $c = (1/2)\log(2\pi e)$.

Alternatively, these gray levels can be expressed in terms of the mean gray level $\overline{g_i}$ across images and the deviations $g_i^j - \overline{g_i}$ from this average in each individual image. The cost of describing the means, can be approximated by

$$\overline{C} = M(\log \overline{\sigma} + c) \tag{2}$$

where $\overline{\sigma}$ is the measured variance of the mean gray levels. Similarly the coding length of describing deviations from the mean is given by

$$C_j^d = M(\log \sigma_j^d + c) \tag{3}$$

where σ_j^d is the measured variance of those deviations in image j. Note that, because we describe the mean across the images, we need only describe $N-1$ of the C_j^d. The description of the Nth one is implicit.

The MDL score is the difference between these two coding lengths, normalized by the number of samples, that is

$$Loss = \overline{C} + \sum_{1\le j\le N-1} C_j^d - \sum_{1\le j\le N} C_j \quad . \tag{4}$$

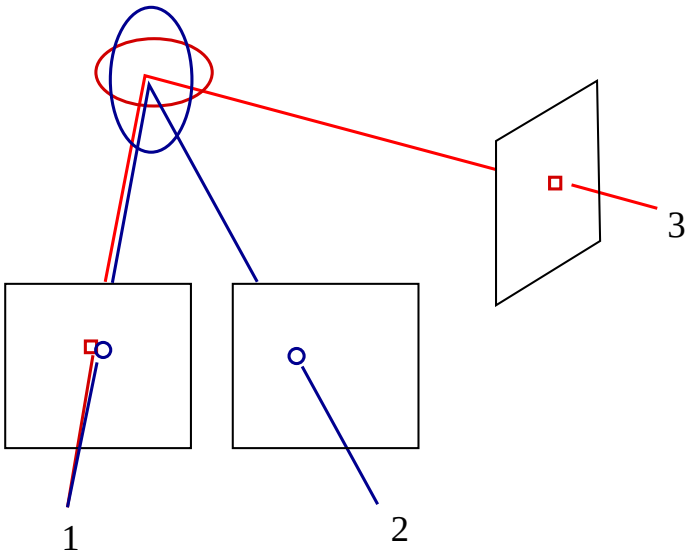


Figure 4: The 3-D statistics of differences depend on the geometric configurations of the cameras. It is necessary to normalize this effect out.

When there is a good match between images, the $g^j_{i\,1\leq j\leq N}$ have a small variance. Consequently the C^d_j should be small, $\overline{C}$ should be approximately equal to any of the C_j and $Loss$ should be negative. However, C_j can only be strongly negative if these costs are large enough, that is, if there is enough texture for a reliable match.

4 Projection Normalization

4.1 The Mahalanobis distance

Assuming that the contribution of each individual match to the statistics is the same ignores many imaging factors like the geometric configuration of the cameras and their resolution, or the distance of the 3D point from the cameras, as illustrated in Fig. 4. There is a simple way to take into account all of these factors, applying a normalization which make the statistics invariant to these imaging factors. In addition, this mechanism makes it possible to take into account the uncertainty in camera parameters, by including them into the observation parameters.

We assume that the observation error (due to image noise and digitalization effects) is Gaussian. This makes it possible to compute the covariance of the reconstruction given the covariance of the observations. Let us consider two reconstructed estimates of a 3-D point, M_1 and M_2 to be compared, and their computed covariance matrices Λ_1 and Λ_2. We weight the squared Euclidean distance between M_1 and M_2 by the sum of their covariances. This yields the squared *Mahalanobis distance*: $(\mathbf{M}_1 - \mathbf{M}_2)^T(\Lambda_1 + \Lambda_2)^{-1}(\mathbf{M}_1 - \mathbf{M}_2)$.

4.2 Determining the reconstruction and reprojection covariances

If the measurements are modeled by the random vector $\mathbf{x}$, of mean $\mathbf{x}_0$ and of covariance $\Lambda_{\mathbf{x}}$, then the vector $\mathbf{y} = f(\mathbf{x})$ is a random vector of mean is $f(\mathbf{x}_0)$ and, up to the first order, covariance $\mathbf{J}_f(\mathbf{x}_0)\Lambda_{\mathbf{x}}\mathbf{J}_f(\mathbf{x}_0)^T$, where $\mathbf{J}_f(\mathbf{x}_0)$ is the Jacobian matrix of f, at the point $\mathbf{x}_0$.

In order to determine the 3-D distribution error in reconstruction, the vector $\mathbf{x}$ is defined by concatenating the 2-D coordinates of each point of the match, ie $[x_1, y_1, x_2, y_2, \ldots x_n, y_n]$ and the result of the function is the 3-D coordinates X, Y, Z of the point M reconstructed from the match, in the least-squares sense. The key is that M is expressed by a closed-form

formula of the form $\mathbf{M} = (\mathbf{L}^T\mathbf{L})^{-1}\mathbf{L}^T\mathbf{b}$, where $\mathbf{L}$ and $\mathbf{b}$ are a matrix and vector which depend on the projection matrices and coordinates of the points in the match. This makes it possible to obtain the derivatives of M with respect to the $2n$ measurements $w_i, i = 1 \ldots n, w = x, y$. We also assume that the errors at each pixel are independent, uniform, and isotropic. The covariance matrix $\Lambda_{\mathbf{x}}$ is then diagonal, therefore each element of Λ_M can be computed as a sum of independent terms for each image.

The above calculations are exact when the mapping between the vector of coordinates of m_i and M (resp. m'_j and M') is linear, since it is only in that case that the distribution of M and M' is Gaussian. The reconstruction operation is exactly linear only when the projection matrices are affine. However, the linear approximation is expected to remain reasonable under normal viewing conditions, and to break down only when the projection matrices are in configurations with strong perspective.

4.3 Simulations

In order to gain insight into the nature of the normalized self-consistency distributions, we investigate the case when the noise in point localization is Gaussian.

We first derive the analytical model for the self-consistency distribution in that case. We then show, using monte-carlo experiments that, provided that the geometrical normalization described in Sec.4 is used, the experimental self-consistency distributions fit this model quite well when perspective effects are not strong. A consequence of this result is that under the hypothesis that the error localization of the features in the images is Gaussian, the self-consistency distribution could be used to recover exactly the accuracy distribution.

Modeling the Gaussian self-consistency distributions. The squared Mahalanobis distance in 3D follows a chi-square distribution with three degrees of freedom:

$$\chi^2_3 = \frac{1}{\sqrt{2\pi}}\sqrt{x}e^{-x/2}$$

In our model, the Mahalanobis distance is computed between M, M', reconstructions in 3D, which are obtained from matches m_i, m'_j of which coordinates are assumed to be Gaussian, zero-mean and with standard deviation σ. If M, M' are obtained from the coordinates m_i, m'_j with a linear transformation A, A', then the covariances are $\sigma^2 AA^T$, $\sigma^2 A'A'^T$. The Mahalanobis distance follows the distribution:

$$d_3 = x^2/\sigma^3\sqrt{2/\pi}e^{-x^2/2\sigma^2} \tag{5}$$

Using the Mahalanobis distance, the self-consistency distributions should be *statistically* independent of the 3D points and projection matrices. Of course, if we were just using the Euclidean distance, there would be no reason to expect such an independence.

Comparison of the normalized and unnormalized distributions To explore the domain of validity of the first-order approximation to the covariance, we have considered three methods to generate random projection matrices:

1. General projection matrices are picked randomly.
2. Projection matrices are obtained by perturbing a fixed, realistic matrix (which is close to affine). Entries of this matrix are each varied randomly within 500% of the initial value.
3. Affine projection matrices are picked randomly.

Each experiment in a set consisted of picking random 3D points, random projection matrices according to the configuration previously described, projecting them, adding random Gaussian noise to the matches, and computing the self-consistency distributions by labelling the matches so that they are perfect.

To illustrate the invariance of the distribution that we can obtain using the normalization, we performed experiments where we computed both the normalized version and the unnormalized version of the self-consistency. As can be seen in Fig. 5, using the normalization reduced dramatically the spread of the self-consistency curves found within each experiment in a set. In particular, in the two last configurations, the resulting spread was very small, which indicates that the geometrical normalization was successful at achieving invariance with respect to 3D points and projection matrices.

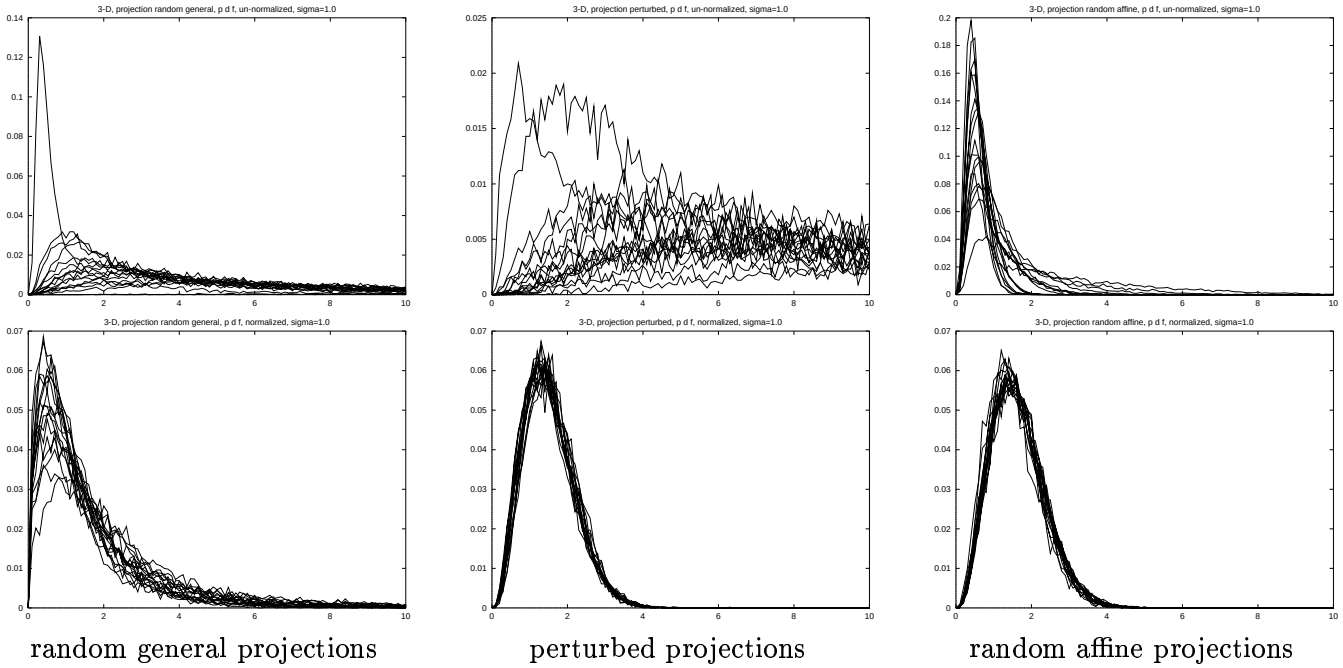


Figure 5: Simulations comparing un-normalized (top) vs normalized (bottom) self-consistency distributions.

Comparison of the experimental and theoretical distributions Using the Mahalanobis distance, we then averaged the density curves within each set of experiments, and tried to fit the model described in Eq. 5 to the resulting curves, for six different values of the standard deviation, $\sigma = 0.5, 1, 1.5, 2, 2.5, 3$. As illustrated in Fig. 6, the model describes the average self-consistency curves very well when the projection matrices are affine (as expected from the theory), but also when they are obtained by perturbation of a fixed matrix. When the projection matrices are picked totally at random, the model does not describe the curves very well, but the different self-consistency curves corresponding to each noise level are still distinguishable.

5 Experiments in measuring the self-consistency of stereo algorithms

5.1 Comparing Two Algorithms

The experiments described here and in the following section are based on the application of stereo algorithms to seventeen scenes, each comprising five images, for a total of 85 images and 170 image pairs. At the highest resolution, each image is a window of about 900 pixels on a side

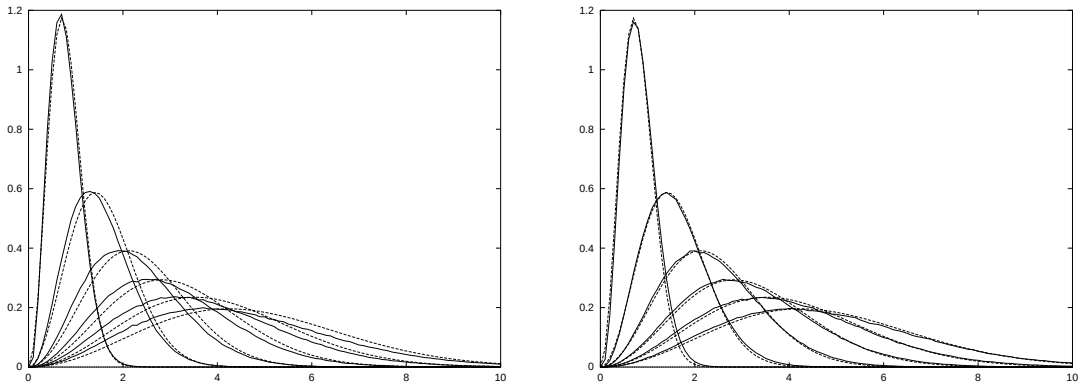


Figure 6: Simulations comparing averaged theoretical (solid) and experimental (dashed) curves.

from images of about 9000 pixels on a side. Some of the experiments were done on Gaussian-reduced versions of the images. These images were controlled and bundle-adjusted to provide accurate camera parameters. The images had a ground resolution of approximately 15cm. A single self-consistency distribution for each algorithm was created by merging the scatter data for that algorithm across all seventeen scenes.

In the top of Fig. 7 we see several representations of the common-xy-coordinate self-consistency distribution described in Sec. 2.2 obtained by applying the simple point-by-point stereo algorithm [10]. The bar graph of Fig. 7(b) is the histogram of the normalized difference in the coordinate of the triangulation of all common-xy-coordinate match pairs. One can see from this histogram that the mode of the differences is about 1 normalized unit. The curve is the integral of this graph, or the cumulative distribution function. One can see from this that about 90% of the match pairs have normalized differences below 2 units. Each point in Figure 1(a) corresponds to a common-xy-coordinate match pair. The x-coordinate of the point in the diagram is the larger of the scores for the two matches, and the y-coordinate is the normalized difference between their triangulated coordinates. The curve in Fig. 7(a) is the 99% significance level. Note that significance level increases as the score increases, indicating that matches with larger scores are less self-consistent than matches with a lower score, an indication of the quality of our MDL score. The drop that we observe for positive values of the score is due to the fact that there are only few common-xy-coordinate match pairs with positive scores, so that calculations done with those values are not statistically meaningful.

In the bottom of Fig. 7 we see the common-xy-coordinate self-consistency distribution for the deformable mesh algorithm [11] applied to the same images. Note that it is significantly more self-consistent than the distribution for the stereo algorithm. This is as expected, since the deformable mesh algorithm was specifically designed to provide highly accurate reconstructions of terrain.

Comparing these two graphs shows some interesting differences between the two algorithms. The deformable mesh algorithm clearly has more outliers (matches with normalized distances above 1), but has a much greater proportion of matches with distances below 0.25. This is not unexpected since the strength of the deformable meshes is its ability to do very precise matching between images. However, the algorithm can get stuck in local minima. Self-consistency now allows us to quantify how often this happens.

But this comparison also illustrates that one must be very careful when comparing algorithms or assessing the accuracy of a given algorithm. The distributions we get are very much dependent on the scenes being used (as would also be the case if we were comparing the algorithms against ground truth—the "gold standard" for assessing the accuracy of a stereo algorithm). In general, the distributions will be most useful if they are derived from a well-defined class of scenes. It

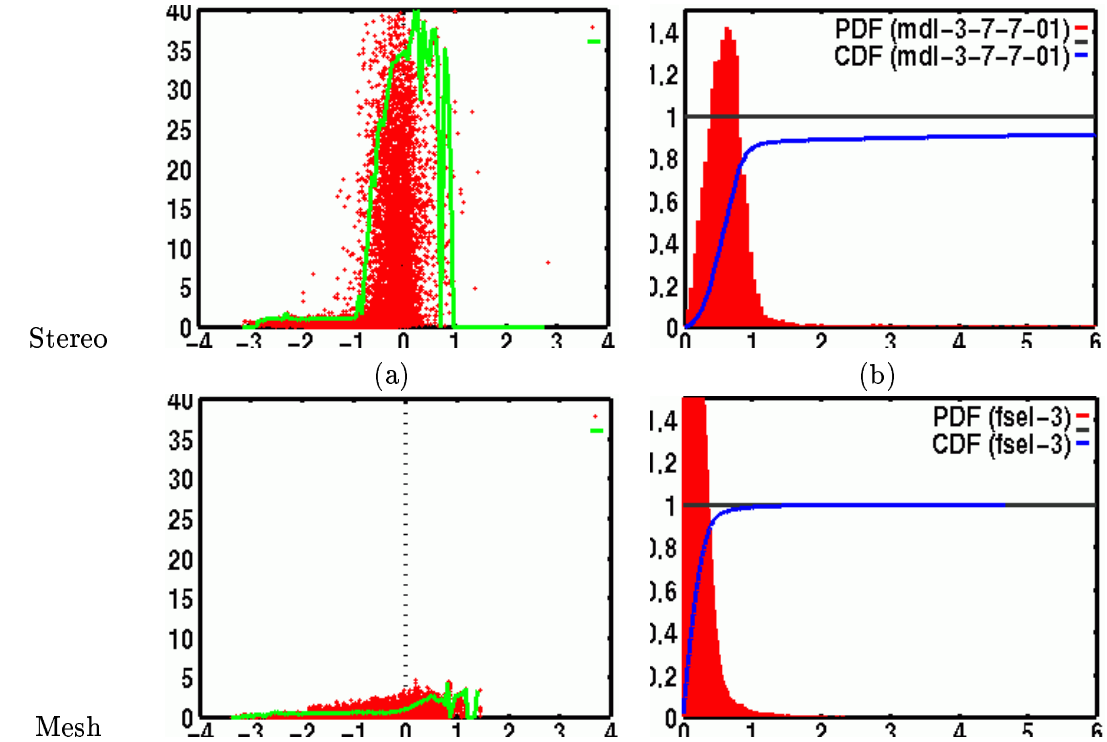


Figure 7: Experimental comparison of two algorithms: point-by-point stereo algorithm (top) vs deformable mesh algorithm (bottom). (a),(c): Scatter diagrams for 170 image pairs of rural scenes. The superimposed curve is the 99% significance level, which is the largest normalized z difference for which 99% of the common-point matches with a given score lie below. (b),(d): The histogram of the normalized z differences for all common-point matches. The superimposed curve is the cumulative distribution of the histogram.

might also be necessary to restrict the imaging conditions (such as resolution or lighting) as well, depending on the algorithm. Only then can the distribution be used to predict the accuracy of the algorithm when applied to images of similar scenes.

5.2 Comparing Three Scoring Functions

To eliminate the dependency on scene content, we propose to use a score associated with each match, as described in Sec. 2.3. We saw scatter diagrams in Figure 2(c) that illustrated how a scoring function might be used to segregate matches according to their expected self-consistency.

In this section we will compare three scoring functions, one based on Minimum Description Length Theory (the MDL score, Sec. 3), the traditional sum–of–squared–differences (SSD) score, and the SSD score normalized by the localization covariance (SSD/GRAD score) [9]. All scores were computed using the same matches computed by our deformable mesh algorithm applied to all image pairs of the seventeen scenes mentioned above. The scatter diagrams for all of the areas were then merged together to produce the scatter diagrams show in Figure 8.

The MDL score has the very nice property that the confidence interval (as defined earlier) rises monotonically with the score, at least until there is a paucity of data, when then score is greater than 2. It also has a broad range of scores (those below zero) for which the normalized distances are below 1, with far fewer outliers than the other scores. The SSD/GRAD score also increases monotonically (with perhaps a shallow dip for small values of the score), but only over a small range.

The traditional SSD score, on the other hand, is distinctly not monotonic. It is fairly non-self-consistent for small scores, then becomes more self-consistent, and then rises again.

5.3 Comparing Window Size

One of the common parameters in a traditional stereo algorithm is the window size. Figure 9 presents one image from six urban scenes, where each scene comprised four images. Figure 10 shows the merged scatter diagrams (a) and global self-consistency distributions (b) for all six scenes, for three window sizes (7×7, 15×15, and 29×29). Some of the observations to note from these experiments are as follows.

First, note that the scatter diagram for the 7×7 window of this class of scenes has many more outliers for scores below -1 than were found in the scatter diagram for the terrain scenes. This is reflected in the global self-consistency distribution in (b), where one can see that about 10% of matches have normalized distances greater than 6. The reason for this is that this type of scene has significant amounts of repeating structure along epipolar lines. Consequently, a score based only on the quality of fit between two windows (such as the MDL-based score) will fail on occasion. A better score would include a measure of the uniqueness of a match along the epipolar line as a second component. We are currently exploring this.

Second, note that the number of outliers in both the scatter diagram and the self-consistency distributions decreases as window size decreases. Thus, large window sizes (in this case) produce more self-consistent results. But it also produces fewer points. This is probably because this stereo algorithm uses left-right/right-left equality as a form of self-consistency filter.

We have also visually examined the matches as a function of window size. When we restrict ourselves to matches with scores below -1, we observe that matches become sparser as window size increases. Furthermore, it appears that the matches are more accurate with larger window sizes. This is quite different from the results of Faugeras *et al.*— [8]. There they found that, in general, matches became denser but less accurate as window size increased. We believe that this is because an MDL score below -1 keeps only those matches for which the scene surface is approximately fronto-parallel within the extent of the window, which is a situation in which larger window sizes increases accuracy. This is borne out by our visual observations of the matches. On the other hand, this result is basically in line with the results of Szeliski and Zabih [28, 30], who show that prediction error decreases with window size.

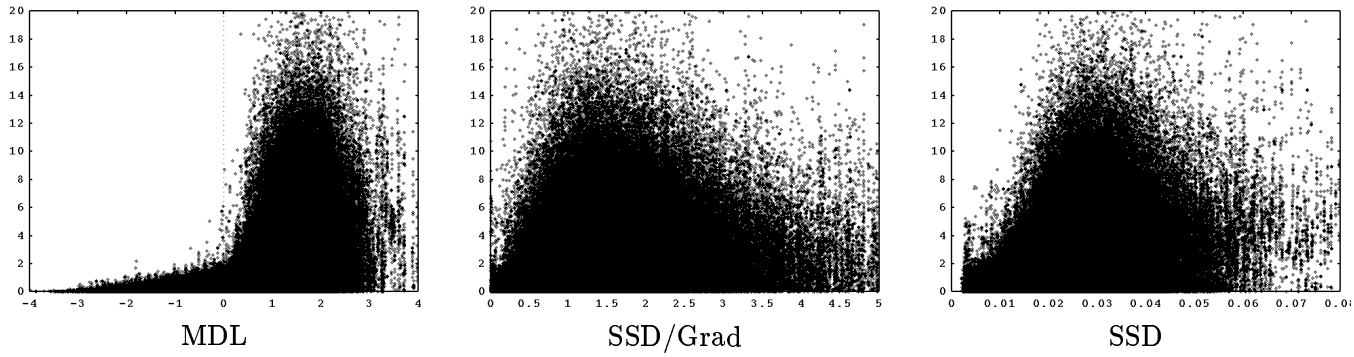


Figure 8: Scatter diagrams for three different scores.

Figure 9: Three of six urban scenes used for the window comparisons. Each scene contained 4 images.

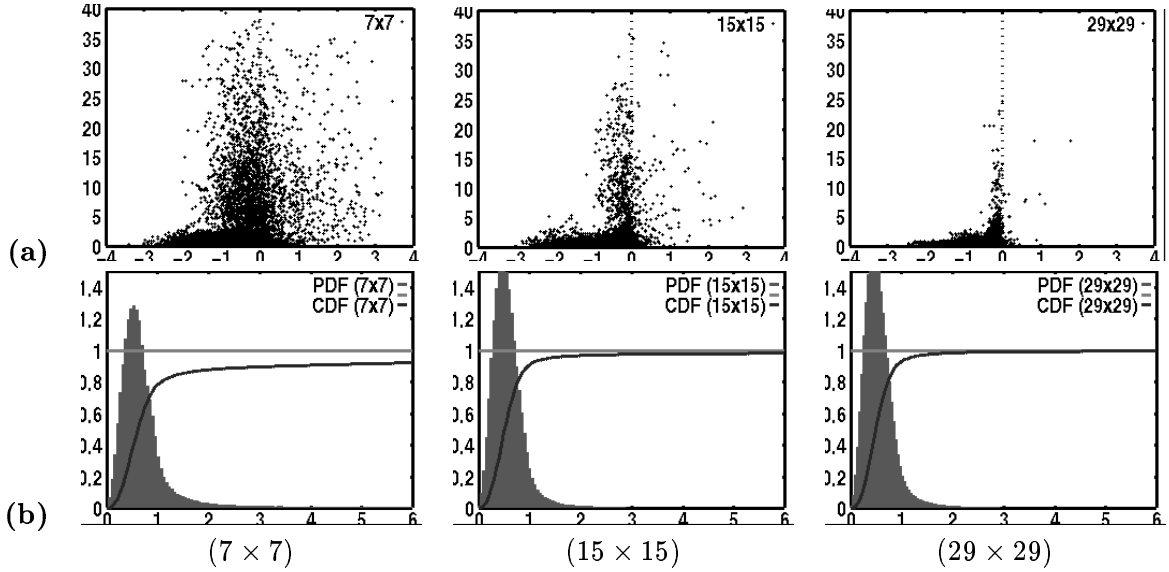


Figure 10: Comparing three window sizes. (a) The combined self-consistency distributions of six urban scenes for window sizes 7×7, 15×15, and 29×29. (b) The scatter diagrams for the MDL score for these urban scenes.

6 Experiments in Change Detection

6.1 The principle

So far, we have assumed that all the images have been taken at exactly the same time, or equivalently, that the scene is static.

One application of the self-consistency distribution is detecting changes in a scene over time. Given two collections of images of a scene taken at two points in time, we can compare matches (from different times) that belong to the same surface element to see if the difference in triangulated coordinates exceeds some significance level. This gives a mechanism for distinguishing changes which are significant from changes which are due to modelization uncertainty.

We would like to be able to compare a reconstruction of a scene created with as few as two images of the scene taken at time against a reconstruction of the same scene created with as few as two images taken at a different time. There is not enough data in these conditions to compute the self-consistency distribution. However, using sets of images of similar scenes, the conditionalization method described in Sec. 2.3 based on the MDL score let us can summarize self-consistency, and therefore expected variation for individual surface elements. As it was found in Sec. 5.2, the MDL score has a stronger correlation with self-consistency than other scores we have examined, and its use is essential for the proposed change detection method to work.

Although our methodology for change detection is quite general, the experiments we conducted are based on two simplifying assumptions. First, we use a specific class of objects: terrain (both rural and urban) viewed from above using aerial imagery. We take advantage of the special nature of terrain to simplify the problem. The 3-D shape is modeled as a single-valued function, whose value represents elevation above the ground plane. Second, we assume that all the camera parameters for all the images are known in a common coordinate system, which we obtain by bundle adjustment over all the images. Together, these two assumptions reduce the problem of detecting changes in 3-D shape to that of finding point-by-point significant differences in scalar values.

6.2 The algorithm

In the first stage, we run the stereo algorithm on a large number of subsets of images of the same class as those in which we want to perform change detection. We use a bucketing method to find all the common-xy-coordinate matches (pairs of matches for which the 3-D reconstruction has the same (x,y) value within a threshold). Each such pair is accumulated in a scatter diagram (see Fig. 7) in which the x-coordinate of is the larger of the scores for the two matches, and the y-coordinate is the normalized difference between their triangulated coordinates. We then extract the significance level curves for the values of significance s% (or in other words, percent confidence in the significance of the change) which we plan to use. For a given value of the score (the x-axis), this is the normalized difference below which s% of the common-xy-coordinate match pairs with that score lie.

In the second stage, we use the pre-computed significance level curves to judge whether a pair of matches derived from images taken at different times is significantly different. We find, using the same technique as before, the common-xy-coordinate matches where each match originates from a different instant, compute the larger of their scores and the normalized difference between triangulated z coordinates. If, for that score, this distance is above the significance level s% then the pair of matches is deemed to be a difference significant with confidence s%.

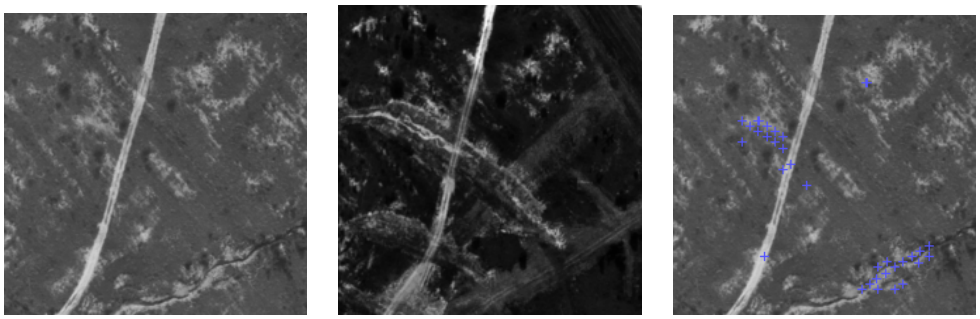
All this assumed that the registration of all images between the two instants is perfect. However, often there is a small registration error in the z-coordinates. For example, in the rural scenes examples, we have often found an error of about half of a meter. Since the changes that we try to detect are of a comparable magnitude, the registration error would cause a severe bias in the results. To solve this problem, once the common-xy-coordinate matches are found, we estimate the registration error between each pair of views taken at a different instant as the

offset in z which minimizes the median of the squared differences between points in each of the views. Because the median is a robust operator, this will give an accurate estimation of the registration error as long as a majority of points didn't have a change in z coordinates.

6.3 Experimental results

Rural scenes

In Fig. 11 we show the changes detected in one of the rural scenes mentioned above, using the deformable mesh algorithm. In Fig. 11 we see one of 5 images of the scene taken in 1995. The dark diagonal near the center of the image is a dried creek bed. In Fig. 11 we see one of 5 images of the same area taken in 1998. The dried creek bed has been filled in with dirt, creating a change in elevation of about 1 meter. We applied the deformable mesh algorithm to one pair of images taken in 1995. We then compared this to the deformable mesh derived from one pair of images taken in 1998. Vertices that were deemed to be significantly different (above the 99% level of the self-consistency distribution of Fig. 7(c)), are overlaid as white cross on the image in Fig. 11 (c), which is a magnified view of the dried creek bed of Fig. 11 (a). We have also applied our algorithm to forested areas of the same rural scene. Although the normalized differences in z-coordinates is sometimes much larger (10 meters), no changes were deemed significant. Indeed, it is known that the mesh algorithm performs poorly on images of tree canopies, so that reconstruction noise could account for the differences.



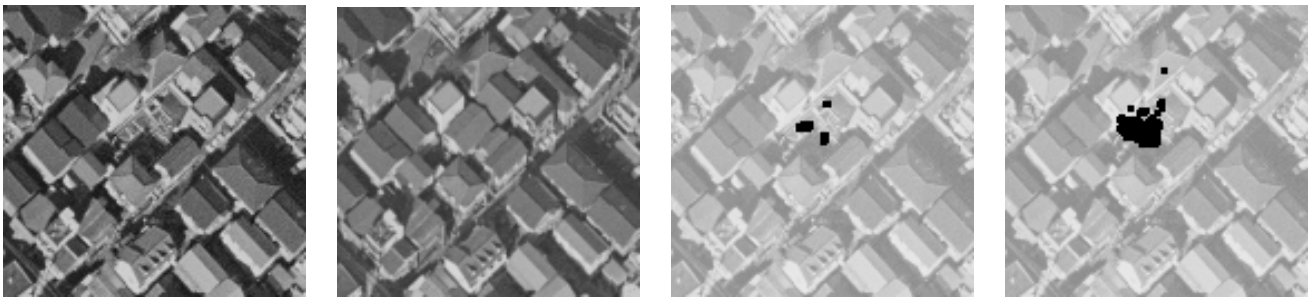
(a) (b) (c)

Figure 11: Changes detected in one of the 17 rural scenes. (a) One of the 5 images of a dried creek bed taken in 1995. (b) One of 5 images of the same area taken in 1998. Note that the dried creek bed has been filled in with dirt, causing a change in elevation of about 1 meter. (c) The significant differences found between the deformable-mesh model created with one pair of 1995 image and a model created using one pair of 1998 images, using the self-consistency distribution of Fig. 7(c)

Urban scenes In Fig. 12 we show the changes (significant differences in z) detected in one of the urban scenes mentioned above, but this time using the stereo algorithm with 15×15 windows, for which the distribution was shown in Fig. 10. In Fig. 12 (a), we see one of 4 images taken at time 1. Note the new building near the center of the image. In Fig. 12 (b) we see one of the images taken at time 2. In Fig. 12 (c) we see the significant differences between the matches derived from a single pair of images taken at time 1 and the matches derived from a single pair of images taken at time 2, for a significance level of 99.99%. In Fig. 12 (d) we have merged the significant differences between each pair of images at time 1 and each pair of images at time 2. Note that virtually all differences are at the location of the new building. For comparison, we show what would happen if we simply thresholded the normalized difference in triangulated z coordinates. In Fig. 13 (a) we show the differences between a single pair of images at time 1

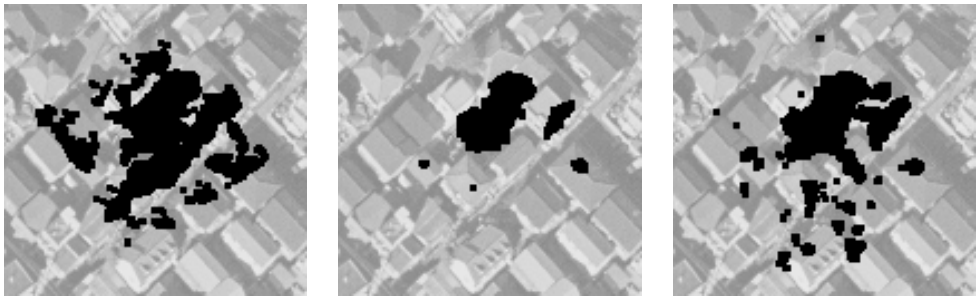
and a single pair of images at time 2, for a threshold of 3 units. In Fig. 13 (b) we show the differences for a threshold of 6 units, which is the average difference found in Fig. 12. This value of the threshold is the highest one for which no correct changes are missed, yet it is seen that many incorrect changes are still detected. In Fig. 13 7(c) we see the union of the differences for all image pairs.

In Fig. 14 and Fig. 15 we see the results of change detection for two other urban scenes, one with a new building, the other without significant changes.



(a) (b) (c) (d)

Figure 12: The first urban scene. (a) Image at time 1. (b) Image at time 2. Note the new buildings near the center of the image. (c) The significant differences between matches derived from one pair of images taken at time 1 and matches derived from one pair of images taken at time 2. (d) The union of all significant differences found between matches derived from all pairs of images taken at time 1 and all pairs of images taken at time 2.



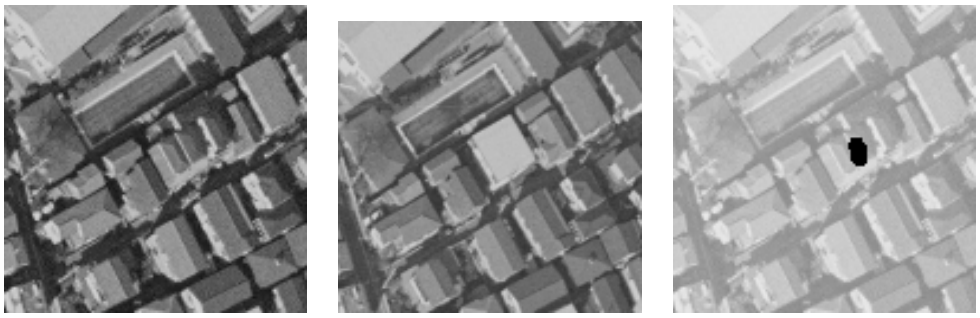
(a) (b) (c)

Figure 13: For comparison, a simple thresholding of the normalized difference in z coordinates. (a) Matches with normalized z differences ¿ 3 units, for the same matches as in Fig. 12. (b) Same as (a), for z differences ¿ 6 units (which is the average difference found in Fig. 12. (c) The union of all differences for all pairs of images, as in Fig. 12 (d).

7 Comparison with Previous work

7.1 Other measures of uncertainty

Existing work on estimating uncertainty without ground truth falls into three categories: analytical, statistical, and empirical approaches.



(a) (b) (c)

Figure 14: A second urban scene. (a) one of the 4 images taken at time 1. (b) One of the 3 images taken at time 2. Note the changed building near the center. (c) The union of all significant differences found between matches derived from all pairs of images taken at time 1 and all pairs of images taken at time 2.

(a) (b) (c)

Figure 15: A third urban scene. (a) one of the 4 images taken at time 1. (b) One of the 3 images taken at time 2. Note that there are no obvious changes in this case. near the center. (c) The union of all significant differences found between matches derived from all pairs of images taken at time 1 and all pairs of images taken at time 2. No changes were detected.

The analytical approaches are based on the idea of error propagation [33]. When the output is obtained by optimizing a certain criterion (like a correlation measure), the shape of the optimization curve [9, 20, 14] or surface [1] provides estimates of the covariance through the second-order derivatives. These approaches make it possible to compare the uncertainty of different outputs given by the same algorithm. However, it is problematic to use them to compare different algorithms.

Statistical approaches make it possible to compute the covariance given only one data sample and a black-box version of an algorithm, by repeated runs of the algorithm, and application of the law of large numbers [7].

Both of the above approaches characterize the performance of a given output only in terms of its expected variation with respect to additive white noise. In [31], the accuracy was characterized as a function of image resolution. The bootstrap methodology [6] goes further, since it makes it possible to characterize the accuracy of a given output with respect to IID noise of unknown distribution. Even if such an approach could be applied to the multiple image correspondence problem, it would characterize the performance with respect to IID sensor noise. Although this is useful for some applications, for other applications it is necessary to estimate the expected accuracy and reliability of the algorithms as viewpoint, scene domain, or other imaging conditions are varied. This is the problem we seek to address with the self-consistency methodology.

Our methodology falls into the realm of empirical approaches. See [25], for a good overview of such approaches.

Szeliski [28] has recently proposed prediction error to characterize the performance of stereo and motion algorithms. Prediction error is the difference between a third real image of a scene and a synthetic image produced from the disparities and known camera parameters of the three images. This approach is especially useful when the primary use of stereo is for view interpolation, since the metric they propose directly measures how well the algorithm has interpolated a view compared to a real image of that same view. In particular, their approach does not necessarily penalize a stereo algorithm for errors in constant-intensity regions, at least for certain viewpoints. Our approach, on the other hand, attempts to characterize self-consistency for all points. Furthermore, our approach attempts to remove the effects of camera configuration in computing the measure over many observations and scenes.

Szeliski and Zabih have recently applied this approach to comparing stereo algorithms [28, 30]. A comprehensive comparison of our two methodologies applied to the the same algorithms and same datasets should yield interesting insights into these two approaches.

An important item to note about our methodology is that the projection matrices for all of the images are provided and assumed to be correct (within their covariances). Thus, we assume that a match produced by the stereo algorithm always lies on the epipolar lines of the images. Consequently, a measure of how far matches lie from the epipolar line, is not relevant.

7.2 Change detection

Change detection is an important task in computer vision that has been addressed early at the image intensity level [22, 19] where it is still a topic of interest [24] with many other papers in between). However, comparing intensity values is not very effective because such changes don't necessarily reflect actual changes in shape, but could be caused by changes in viewing and illumination conditions or even in reflectance (e.g., seasonal changes). Although it has been attempted, this is not easy to take them into account at this level. For man-made objects such as buildings, higher-level comparisons have been proposed, based on feature organization [26], and 3-D models [4, 13] These specialized approaches are the most successful, but are not applicable to more general objects like natural terrain. A few of the ideas needed for general change detection in shape are found in other areas of computer vision. In work on tracking (see for instance [5] which deals with small bodies in a natural environment), statistics have been computed during a learning phase and then used to differentiate between significant and insignificant changes.

Of course, the problem is simplified by the fact that the camera is stationary, whereas we want to deal with various viewpoints. In mobile robotics (see for instance [2]), the problem of fusing several general 3-D maps to maintain representations of the environment of a robot has been cast in a statistically rigorous framework taking into account uncertainties. However, the specific issue of change detection has not been addressed there.

8 Extending the Self-Consistency paradigm

So far we have studied the self-consistency of stereo algorithms. In this section, we discuss a more general methodology to measure the self-consistency of computer vision algorithms.

8.1 A Formalization of Self-Consistency

We give a simple formalization of a computer vision algorithm as a function that takes an observation Ω of a world W as input and produces a set of hypotheses H about the world as output:

$$H = (h_1, h_2, \ldots, h_n) = F(\Omega, W).$$

An observation Ω is one or more images of the world taken at the same time, perhaps accompanied by meta-data, such as the time the image(s) was acquired, the internal and external camera parameters, and their covariances.

A hypothesis h nominally refers to some aspect or element of the world (as opposed to some aspect of the observation), and it nominally estimates some attribute of the element it refers to. We formalize this with the following set of functions that depend on both F and Ω:

1. $Ref(h)$, the referent of the hypothesis h (i.e., which element in the world that the hypothesis refers to).
2. $R(h, h') = Prob(Ref(h) = Ref(h')$, an estimate of the probability that two hypotheses h and h', (computed from two observations of the same world), refer to the same object or process in the world.
3. $Att(h)$, an estimate of some well-defined attribute of the referent.
4. $Acc(h)$, an estimate of the accuracy distribution of $Att(h)$. When this is well-modeled by a normal distribution, it can be represented implicitly by its covariance, $Cov(h)$.
5. $Score(h)$, an estimate of the confidence that $Att(h)$ is correct.

Intuitively, we can state that two hypotheses h and h', derived from observations Ω and Ω' of a static world W, are consistent with each other if they both refer to the same object in the world and the difference in their estimated attributes is small relative to their accuracies, or if they do not refer to the same object. When the accuracy is well modeled by a normal distribution, the consistency of two hypotheses, $C(h, h')$, can be written as

$$C(h, h') = R(h, h')(Att(h) - Att(h'))^T (Cov(h) + Cov(h'))^{-1} (Att(h) - Att(h'))^T$$

Note that the second term on the right is the Mahalanobis distance between the attributes, which we refer to as the normalized distance between attributes throughout this paper.

Given the above, we can measure the self-consistency of an algorithm as the histogram of $C(h, h')$ over all pairs of hypotheses in $H = F(\Omega(W))$ and $H' = F(\Omega'(W))$, over all observations over all suitable static worlds W. We call this distribution of $C(h, h')$ the self-consistency distribution of the computer vision algorithm F over the worlds W. To simplify the exposition below, we compute this distribution only for pairs h and h' for which $R(h, h') \approx 1$. We will discuss the utility of the full distribution in future work.

8.2 Applications to Self-Consistency of Stereo Algorithms

To help the reader's intuition, in this section we detail how the previous abstract self-consistency formalism applies to stereo algorithms. Stereo algorithms attempt to reconstruct 3D surface geometry from multiple images, assuming that the camera geometry is known.

The hypothesis h produced by a traditional stereo algorithm is a pair of image coordinates $(\mathbf{x_0}, \mathbf{x_1})$ in each of two images, (I_0, I_1). In its simplest form, a stereo match hypothesis h asserts that the closest opaque surface element along the optic ray through $\mathbf{x_0}$ is the same as the closest opaque surface element along the optic ray through $\mathbf{x_1}$. That is, the referent of h, $Ref(h)$, is the closest opaque surface element along the optic rays through both $\mathbf{x_0}$ and $\mathbf{x_1}$.

Consequently, as the camera geometry is varied, two stereo hypotheses should have the same referent if their image coordinates are the same in one image. Self-consistency, in this case, is a measure of how often (and to what extent) this assertion is true.

The above observation can be used to write the following set of associated functions for a stereo algorithm. We assume that all matches are accurate to within some nominal accuracy, σ, in pixels (typically $\sigma = 1$). This can be extended to include the full covariance of the match coordinates.

1. $Ref(h)$, The closest opaque surface element visible along the optic rays through the match points.
2. $R(h, h') = 1$ if h and h' have the same coordinate (within σ) in one image; 0 otherwise.
3. $Att(h)$, The triangulated 3D (or projective) coordinates of the surface element.
4. $Acc(h)$, The covariance of $Att(h)$, given that the match coordinates are $N(\mathbf{x_0}, \sigma)$ and $N(\mathbf{x_0}, \sigma)$ random variables.
5. $Score(h)$, A measure such as normalized cross-correlation or coding loss.

8.3 Applying Self-consistency to other domains

Shape-From-Shading Shape-from-shading algorithms [12] attempt to reconstruct the surface orientation from a single image, assuming that the light source direction is known.

The hypothesis produced by a shade from shading algorithm is a pair of image coordinates and 3-D unit vector $(\mathbf{x}, \mathbf{n})$. This asserts that the surface orientation at the closest opaque surface element along the optic ray through $\mathbf{x}$ is the vector $\mathbf{n}$. As the light source direction is varied, two shape-from-shading hypotheses should have the same surface orientation if their image coordinates are the same in one image.

Line-drawing interpretation Line-drawing interpretation algorithms [27, 18] attempt to build a wire-frame 3-D model from a line-drawing sketch of a polyhedral object viewed from a certain direction.

The hypothesis produced by a wire-frame reconstruction algorithm is a segment label and a reconstructed 3-D segment given by the 3-D coordinates of its end points $(\lambda, [A, B])$. This asserts that an orthographic projection of the 3D segment $[A, B]$ yields the 2D segment of label λ. Corresponding segments across different views are supposed to have the same label, therefore when the direction of projection is varied, two line-drawing interpretation hypotheses should yield the same 3D segment if they have the same label. This criterion was used to assess the results of two different algorithms in [18].

9 Conclusion and Perspectives

We have introduced a general formalization of a perceptual observation called self-consistency. We have proposed a methodology based on this formalization as a means of estimating the accuracy and reliability of point-correspondence algorithms algorithms, comparing different stereo

algorithms, comparing different scoring functions, comparing window sizes, and detecting change over time. We have presented a detailed prescription for applying this methodology to multiple-image point-correspondence algorithms, without any need for ground truth or camera calibration, and have demonstrated it's utility in several experiments.

The self-consistency distribution is a very simple idea that has powerful consequences. It can be used to compare algorithms, compare scoring functions, evaluate the performance of an algorithm across different classes of scenes, tune algorithm parameters (such as window size), reliably detect changes in a scene, and so forth. All of this can be done for little manual cost beyond the precise estimation of the camera parameters and perhaps manual inspection of the output of the algorithm on a few images to identify systematic biases.

Finally, we believe that the general self-consistency formalism developed in Sec. 8.1, which examines the *self-consistency* of an algorithm across independent experimental trials of different "viewpoints" (or imaging parameters) of a static scene, can be used to assess the accuracy and reliability of algorithms dealing with a range of computer vision problems.

Once we can measure the self-consistency of an algorithm, and we observe that this measure remains reasonably constant over many scenes (at least for certain subsets), then we can be reasonably confident that the algorithm will be self-consistent over new scenes. More importantly, such algorithms are also likely to exhibit the self-consistency property of the human visual system: *given a single view of a new scene, such an algorithm is likely to produce inferences that would be self-consistent with other views of the scene should they become available later.* Thus, measuring self-consistency is a critical step towards discovering (and eventually designing) self-consistent algorithms. It could also be used to learn the parameters of an algorithm that lead to self-consistency over a wide range of scenes without the need for external training data or "ground truth."

References

[1] P. Anandan. A computational framework and an algorithm for the measurement of visual motion. *IJCV*, 2:283–310, 1989.

[2] N. Ayache. *Artificial vision for mobile robots.* MIT Press, 1991.

[3] N. Ayache and F. Lustman. Fast and reliable passive trinocular stereovision. In *ICCV*, pages 422–427, 1987.

[4] M. Bejanin, A. Huertas, G. Medioni, and R. Nevatia. Model validation for change detection. In *WACV94*, pages 160–167, 1994.

[5] T. Boult, R. Micheals, A. Erkan, P. Lewis, C. Powers, C. Qian, and W. Yin. Frame-rate multi-body tracking for surveillance. In *DARPA98*, pages 305–313, 1998.

[6] K. Cho, P. Meer, and J. Cabrera. Performance assessment through bootstrap. *PAMI*, 19(11):1185–1198, November 1997.

[7] G. Csurka, C. Zeller, Z. Zhang, and O. Faugeras. Characterizing the uncertainty of the fundamental matrix. *CVGIP-IU*, 1996.

[8] O. Faugeras, P. Fua, B. Hotz, R. Ma, L. Robert, M. Thonnat, and Z. Zhang. Quantitative and Qualitative Comparison of some Area and Feature-Based Stereo Algorithms. In W. Forstner and S. Ruwiedel, editors, *International Workshop on Robust Computer Vision: Quality of Vision Algorithms*, pages 1–26, Karlsruhe, Germany, March 1992.

[9] W. Forstner. On the geometric precision of digital correlation. In *International archives of photogrammetry and remote sensing*, volume 24-III, pages 176–189, Helsinki, 1982.

[10] P. Fua. Combining Stereo and Monocular Information to Compute Dense Depth Maps that Pres erve Depth Discontinuities. In *Int. Joint Conf. on AI*, pages 1292–1298, Sydney, Australia, August 1991.

[11] P. Fua and Y. G. Leclerc. Object-Centered Surface Reconstruction: Combining Multi-Image Stereo and Shading. *IJCV*, 16:35–56, 1995.

[12] B.K.P. Horn and M.J. Brooks. *Shape from Shading*. MIT Press, 1989.

[13] A. Huertas and R. Nevatia. Detecting changes in aerial views of man-made structures. *IVC*, 18(8):583–596, May 2000.

[14] T. Kanade and M. Okutomi. A stereo matching algorithm with an adaptive window: Theory and experiment. *PAMI*, 16(9):920–932, 1994.

[15] Y. Leclerc, Q.-T. Luong, and P. Fua. Self-consistency: a novel approach to characterizing the accuracy and reliability of point correspondence algorithms. In *Proc. Image Understanding Workshop*, pages 793–807, Monterey, CA, 1998.

[16] Y. Leclerc, Q.-T. Luong, and P. Fua. Self-consistency: a novel approach to characterizing the accuracy and reliability of point correspondence algorithms. In *Proceedings of the One-day Workshop on Performance Characterisation and Benchmarking of Vision Systems*, Las Palmas de Gran Canaria, Canary Islands, Spain, 1999.

[17] Y. G. Leclerc. Constructing simple stable descriptions for image partitioning. *Intl. Journal of Computer Vision*, 3(1):73–102, 1989.

[18] Y.G. Leclerc and M.A. Fischler. An optimization-based approach to the interpretation of single line drawings as 3d wire frames. *IJCV*, 9(2):113–136, November 1992.

[19] R.L. Lillestrand. Techniques for change detection. *TC*, 21(7):654–659, July 1972.

[20] L. Matthies. Stereo vision for planetary rovers: Stochastic modeling to near real-time implementation. *IJCV*, 8(1):71–91, July 1992.

[21] R. Mohr, F. Veillon, and L. Quan. Relative 3d reconstruction using multiple uncalibrated images. In *CVPR*, pages 543–548, NYC, 1993.

[22] L.H. Quam. *Computer Comparison of Pictures*. PhD thesis, Stanford University, 1971.

[23] J. Rissanen. Minimum-description length principle. *Encyclopedia of statistical sciences*, 3(1):73–102, 1989.

[24] P.L. Rosin. Thresholding for change detection. In *ICCV98*, pages 274–279, 1998.

[25] W. Førstner. Diagnostics and performance evaluation in computer vision. In *Performance versus Methodology in Computer Vision, NSF/ARPA Workshop*, Seattle, WA, 1994.

[26] S. Sarkar and K.L. Boyer. Quantitative measures of change based on feature organization: Eigenvalues and eigenvectors. *CVIU*, 71(1):110–136, July 1998.

[27] K. Sugihara. Machine interpretation of line drawings. In *MIT Press*, 1986.

[28] R. Szeliski. Prediction error as a quality metric for motion and stereo. In *ICCV99*, Corfu, Greece, September 1999.

[29] R. Szeliski and S.B. Kang. Recovering 3d shape and motion from image streams using nonlinear least squares. *JVCIR*, pages 10–28, 1994.

[30] R. Szeliski and R. Zabih. An experimental comparison of stereo algorithms. In *Proceedings of the Vision Algorithms: Theory and Practice Workshop (ICCV99)*, Corfu, Greece, September 1999.

[31] P.H.S. Torr and A. Zissermann. Performance characterization of fundamental matrix estimation under image degradation. *mva*, 9:321–333, 1997.

[32] M. Yachida, Y. Kitamura, and M. Kimachi. Trinocular vision: New approach for correspondence problem. In *ICPR*, pages 1041–1044, 1986.

[33] S. Yi, R.M. Haralick, and L.G. Shapiro. Error propagation in machine vision. *MVA*, 7:93–114, 1994.